

This Page Is Inserted by IFW Operations
and is not a part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

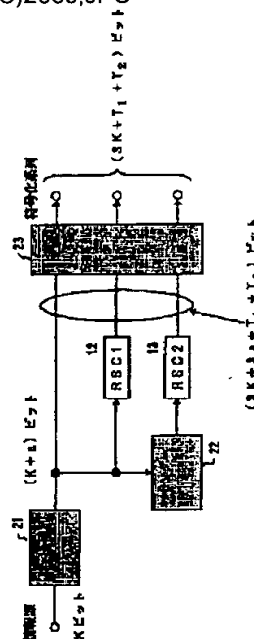
As rescanning documents *will not* correct images,
please do not report the images to the
Image Problem Mailbox.

THIS PAGE BLANK (USPTO)

(43) Date of publication of application: **19.12.00**

COPYRIGHT: (C)2000,JPO

SOLUTION: An input sequence (an output of a bi addition processing part 21 and, for example, 664 bit) is N divided and is written into a two-dimensional arrangement (a buffer). A primitive element of a finite body of a reference number P is obtained and a table in order of its index expression is prepared. The table is made to be an order replacing table to refer to in order to replace orders of data on the first line of two-dimensional buffer. By order replacing table for replacing orders of the data from the first line to the eighth line, orders of the data of a block are replaced. That is, the orders of the data of the block are replaced by the order replacing table.



H03M 13/27
G06F 11/10
H03M 13/13
H03M 13/29

(30) Priority: 19.02.99 JP 11042137
05.04.99 JP 11098160

THIS PAGE BLANK (USPTO)

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2000-353965

(P2000-353965A)

(43) 公開日 平成12年12月19日 (2000. 12. 19)

(51) Int.Cl. ⁷	識別記号	F I	テーマコード* (参考)
H 0 3 M 13/27		H 0 3 M 13/27	
G 0 6 F 11/10	3 3 0	G 0 6 F 11/10	3 3 0 F
H 0 3 M 13/13		H 0 3 M 13/13	
13/29		13/29	

審査請求 未請求 請求項の数20 O L (全 19 頁)

(21) 出願番号 特願2000-42040 (P2000-42040)

(22) 出願日 平成12年2月18日 (2000. 2. 18)

(31) 優先権主張番号 特願平11-42137

(32) 優先日 平成11年2月19日 (1999. 2. 19)

(33) 優先権主張国 日本 (J P)

(31) 優先権主張番号 特願平11-98160

(32) 優先日 平成11年4月5日 (1999. 4. 5)

(33) 優先権主張国 日本 (J P)

(71) 出願人 392026693

株式会社エヌ・ティ・ティ・ドコモ

東京都千代田区永田町二丁目11番1号

(72) 発明者 須田 博人

東京都港区虎ノ門二丁目10番1号 エヌ・

ティ・ティ移動通信網株式会社内

(72) 発明者 渋谷 彰

東京都港区虎ノ門二丁目10番1号 エヌ・

ティ・ティ移動通信網株式会社内

(74) 代理人 100070150

弁理士 伊東 忠彦

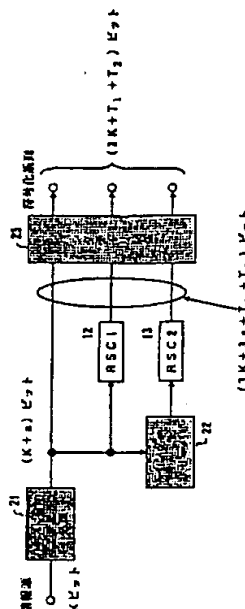
(54) 【発明の名称】 インターリーピング方法、インターリーピング装置、ターボ符号化方法及びターボ符号化装置

(57) 【要約】

【課題】 多様なフレーム長であっても、少ない演算量で効率的に系列のランダム化を実現できるインターリーピングを提供することを目的とするものである。

【解決手段】 素数Pをベースにした長さのブロックを複数個有するデータ系列を入力し、標数がPの有限体の元に所定の演算を行い、その順序を並べ替えて、順序入替えデータを生成し、この順序入替えデータを用いて、入力された前記データ系列のデータの順序を入替える。

本発明の第1の実施例によるターボ符号器を説明するための図



【特許請求の範囲】

【請求項 1】 素数 P をベースにした長さのブロックを複数個有するデータ系列を入力する第 1 の段階と、
標数が P の有限体の元に所定の演算を行い、その順序を並べ替えて、順序入替えデータを生成する第 2 の段階と、
該順序入替えデータを用いて、入力された前記データ系列のデータの順序を入替える第 3 の段階とを有することを特徴とするインターリーピング方法。

【請求項 2】 素数 P を生成又は記録する第 1 の段階と、
入力系列を長さ P の N 個のブロック B_1, B_2, \dots, B_N に分割する第 2 の段階と、
標数が P の有限体の元をその元のべき表現の指数部分の値の順に並べた系列を第 1 の順序入替えデータとして生成又は記録する第 3 の段階と、

($P-1$) とは互いに素な ($N-1$) 個の整数 p_1, p_2, \dots, p_{N-1} を生成又は記録する第 4 の段階と、
第 1 の順序入替えデータ系列を p_i 個飛びに巡回的に読み出して第 i の順序入替えデータを得る処理を $i=1 \sim N-1$ だけ繰り返して第 2 ～第 N の順序入替えデータを生成又は記録する第 5 の段階と、
第 1 から N までの順序入替えデータを用いてブロック B_1, B_2, \dots, B_N 中の順序を入替える第 6 の段階と、
並び替えられた N 個の各ブロックから予め決められた順序により各データを読み出す第 7 の段階とを有することを特徴とするインターリーピング方法。

【請求項 3】 素数 P を生成又は記録する第 1 の段階と、
入力系列を長さ P の N 個のブロック B_1, B_2, \dots, B_N に分割する第 2 の段階と、
標数が P の有限体の元をその元のべき表現の指数部分の値の順に並べた系列を生成又は記録する第 3 の段階と、
このべき表現に用いた原始元とは互いに素な N 個の整数 q_1, q_2, \dots, q_N を生成又は記録する第 4 の段階と、
第 0 の順序入替えデータ系列の各データに q_i を法 P で加え得られた元のべき表現の指数部分の値の系列を第 i の順序入替えデータとする処理を $i=1 \sim N$ まで繰り返して第 1 ～第 N の順序入替えデータを生成する第 5 の段階と、
第 1 から N までの順序入替えデータを用いてブロック B_1, B_2, \dots, B_N 中のデータの順序を入替える第 6 の段階と、
並べ替えられた N 個の各ブロックから予め決められた順序により各データを読み出す第 7 の段階とを有することを特徴とするインターリーピング方法。

【請求項 4】 素数 P を生成又は記録する第 1 の段階

と、

入力系列を長さ ($P-1$) の N 個のブロック B_1, B_2, \dots, B_N に分割する第 2 の段階と、
標数が P の有限体の元をその元のべき表現の指数部分の値の順に並べた系列の最後尾のデータを削除した系列を生成又は記録する第 3 の段階と、

($P-1$) とは互いに素な ($N-1$) 個の整数 p_1, p_2, \dots, p_{N-1} を生成又は記録する第 4 の段階と、
第 1 の順序入替えデータ系列を p_i 個飛びに巡回的に読み出して第 i の順序入替えデータを得る処理を $i=1 \sim N-1$ だけ繰り返して第 2 ～第 N の順序入替えデータを生成又は記録する第 5 の段階と、
第 1 から N までの順序入替えデータを用いてブロック B_1, B_2, \dots, B_N 中の順序を入替える第 6 の段階と、
並び替えられた N 個の各ブロックから予め決められた順序により各データを読み出す第 7 の段階とを有することを特徴とするインターリーピング方法。

【請求項 5】 素数 P を生成又は記録する第 1 の段階と、

入力系列を長さ ($P+1$) の N 個のブロック B_1, B_2, \dots, B_N に分割する第 2 の段階と、
標数が P の有限体の元をその元のべき表現の指数部分の値の順に並べた系列の最後尾のデータに前記素数を追加した系列を生成又は記録する第 3 の段階と、
($P-1$) とは互いに素な ($N-1$) 個の整数 p_1, p_2, \dots, p_{N-1} を生成又は記録する第 4 の段階と、
第 1 の順序入替えデータ系列を p_i 個飛びに巡回的に読み出して第 i の順序入替えデータを得る処理を $i=1 \sim N-1$ だけ繰り返して第 2 ～第 N の順序入替えデータを生成又は記録する第 5 の段階と、
第 1 から N までの順序入替えデータを用いてブロック B_1, B_2, \dots, B_N 中の順序を入替える第 6 の段階と、
並び替えられた N 個の各ブロックから予め決められた順序により各データを読み出す第 7 の段階とを有することを特徴とするインターリーピング方法。

【請求項 6】 前記第 7 の段階の予め決められた順序は、ターボ符号におけるエラーフロアーの値を基準とすることを特徴とする請求項 2 ないし 5 のいずれか一項記載のインターリーピング方法。

【請求項 7】 前記分割数 N を k (k は 2 以上の整数) 個予め決めておき、第 5 の段階で生成する第 1 から N までの順序入替えデータを k 通り生成して、最も特性の良い分割数の順序入替えデータを用いることを特徴とする請求項 2 ないし 5 のいずれか一項記載のインターリーピング方法。

【請求項 8】 素数 P をベースにした長さのブロックを複数個有するデータ系列を入力する第 1 の手段と、
標数が P の有限体の元に所定の演算を行い、その順序を

並べ替えて、順序入替えデータを生成する第2の手段と、

該順序入替えデータを用いて、入力された前記データ系列のデータの順序を入替える第3の手段とを有することを特徴とするインターリーピング装置。

【請求項9】 素数 P を生成又は記録する第1の手段と、

入力系列を長さ P の N 個のブロック B_1, B_2, \dots, B_N に分割する第2の手段と、

標数が P の有限体の元をその元のべき表現の指数部分の値の順に並べた系列を第1の順序入替えデータとして生成又は記録する第3の手段と、

$(P-1)$ とは互いに素な $(N-1)$ 個の整数 p_1, p_2, \dots, p_{N-1} を生成又は記録する第4の手段と、

第1の順序入替えデータ系列を p_i 個飛びに巡回的に読み出して第 i の順序入替えデータを得る処理を $i=1 \sim N-1$ だけ繰り返して第2～第 N の順序入替えデータを生成又は記録する第5の手段と、

第1から N までの順序入替えデータを用いてブロック B_1, B_2, \dots, B_N 中の順序を入替える第6の手段と、

並び替えられた N 個の各ブロックから予め決められた順序により各データを読み出す第7の手段とを有することを特徴とするインターリーピング装置。

【請求項10】 素数 P を生成又は記録する第1の手段と、

入力系列を長さ P の N 個のブロック B_1, B_2, \dots, B_N に分割する第2の手段と、

標数が P の有限体の元をその元のべき表現の指数部分の値の順に並べた系列を生成又は記録する第3の手段と、

このべき表現に用いた原始元とは互いに素な N 個の整数 q_1, q_2, \dots, q_N を生成又は記録する第4の手段と、

第0の順序入替えデータ系列の各データに q_i を法 P で加え得られた元のべき表現の指数部分の値の系列を第 i の順序入替えデータとする処理を $i=1 \sim N$ まで繰り返して第1～第 N の順序入替えデータを生成する第5の手段と、

第1から N までの順序入替えデータを用いてブロック B_1, B_2, \dots, B_N 中のデータの順序を入替える第6の手段と、

並べ替えられた N 個の各ブロックから予め決められた順序により各データを読み出す第7の段階とを有することを特徴とするインターリーピング装置。

【請求項11】 素数 P を生成又は記録する第1の手段と、

入力系列を長さ $(P-1)$ の N 個のブロック B_1, B_2, \dots, B_N に分割する第2の手段と、

標数が P の有限体の元をその元のべき表現の指数部分の

値の順に並べた系列の最後尾のデータを削除した系列を生成又は記録する第3の手段と、

$(P-1)$ とは互いに素な $(N-1)$ 個の整数 p_1, p_2, \dots, p_{N-1} を生成又は記録する第4の手段と、

第1の順序入替えデータ系列を p_i 個飛びに巡回的に読み出して第 i の順序入替えデータを得る処理を $i=1 \sim N-1$ だけ繰り返して第2～第 N の順序入替えデータを生成又は記録する第5の手段と、

第1から N までの順序入替えデータを用いてブロック B_1, B_2, \dots, B_N 中の順序を入替える第6の手段と、

並び替えられた N 個の各ブロックから予め決められた順序により各データを読み出す第7の手段とを有することを特徴とするインターリーピング装置。

【請求項12】 素数 P を生成又は記録する第1の手段と、

入力系列を長さ $(P+1)$ の N 個のブロック B_1, B_2, \dots, B_N に分割する第2の手段と、

標数が P の有限体の元をその元のべき表現の指数部分の値の順に並べた系列の最後尾に前記素数を追加した系列を生成又は記録する第3の手段と、

このべき表現に用いた原始元とは互いに素な N 個の整数 q_1, q_2, \dots, q_N を生成又は記録する第4の手段と、

第0の順序入替えデータ系列の各データに q_i を法 P で加え得られた元のべき表現の指数部分の値の系列を第 i の順序入替えデータとする処理を $i=1 \sim N$ まで繰り返して第1～第 N の順序入替えデータを生成する第5の手段と、

第1から N までの順序入替えデータを用いてブロック B_1, B_2, \dots, B_N 中のデータの順序を入替える第6の手段と、

並べ替えられた N 個の各ブロックから予め決められた順序により各データを読み出す第7の手段とを有することを特徴とするインターリーピング装置。

【請求項13】 前記第7の手段の予め決められた順序は、ターボ符号におけるエラーフロアーの値を基準とすることを特徴とする請求項9ないし12のいずれか一項記載のインターリーピング装置。

【請求項14】 前記分割数 N を k (k は2以上の整数) 個予め決めておき、第5の手段で生成する第1から N までの順序入替えデータを k 通り生成して、最も特性の良い分割数の順序入替えデータを用いることを特徴とする請求項9ないし12のいずれか一項記載のインターリーピング装置。

【請求項15】 請求項1ないし5の何れか一項記載のインターリーピング方法をターボ符号化装置の内部インターリーピング方法とするターボ符号化方法。

【請求項16】 入力ビット数が予め決められた数のビット数に足りない場合には、これに一致するようにビッ

ト数を増加させる段階と、

符号化されたビット数を前記ビット数を増加させる前のビット数にまで削減する段階を含むことを特徴とする請求項 15 記載のターボ符号化方法。

【請求項 17】 請求項 16 記載のターボ符号化方法において、ビットレベションを用いて前記ビット数を増加させることを特徴とするターボ符号化方法。

【請求項 18】 複数の符号化器と、請求項 8 ないし 12 の何れか一項記載のインターリーピング装置とを具備するターボ符号化装置。

【請求項 19】 入力ビット数が予め決められた数のビット数に足りない場合には、これに一致するようにビット数を増加させる手段と、
符号化されたビット数を前記ビット数を増加させる前のビット数にまで削減する手段を含むことを特徴とする請求項 18 記載のターボ符号化装置。

【請求項 20】 請求項 19 記載のターボ符号化装置において、ビットレベションを用いて前記ビット数を増加させることを特徴とするターボ符号化装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、バースト誤りに対して有効なターボ符号化技術に係り、特に間引き処理を行わないか、又は行っても僅かなビット数に限定され、しかも演算量を軽減したインターリーピング方法、インターリーピング装置、ターボ符号化方法及びターボ符号化装置に関する。

【0002】本発明は、ディジタル伝送やディジタル記録など誤り訂正符号を用いて通信の信頼性を上げることが要求される分野で応用され、特にマルチメディアのように通信のフレキシビリティが必要な分野で有効である。

【0003】

【従来の技術】近年提案された能力の高い誤り訂正符号を用いるターボ符号器は複数の符号器で構成されており、各符号器間の冗長系列の相関性を少なくするためにインターリーブ（インターリーピング処理を行う手段）を介して各符号器間が接続されている。このインターリーブは、ターボ符号の能力を決定する大変重要なものとなっている。

【0004】図 1 (a)、(b) は、ターボ符号化器の構成例を示す図である。図 1 (a) に示すように、ターボ符号化器は、複数の再帰的組織畳み込み符号化器 (RSC1) 12、(RSC2) 13 と、インターリーブ 11 とを具備して構成されている。各再帰的組織畳み込み符号化器 12、13 は、図 1 (b) に示すように、加算器 14、15 と単位遅延素子 (D) 16、17 が図示するように接続されて構成されている。図 1 (a) に示されている例のように、ターボ符号化器は入力 d (K ビット) に対して、出力 $X1 \sim X3$ を符号化系列として出力

している。ここで、冗長ビット $X1$ と $X2$ との相関性を少なくするために、再帰的組織畳み込み符号化器 (RSC2) 13 の前にインターリーブ 11 を挿入している。また、図 1 (c) に示すように、ターボ復号器は 2 つのデコーダ 1、2、2 つのインターリーブ 3、4、及びインターリーブの逆の処理を行うデインターリーブ 5 から構成されている。

【0005】なお、デジタル・システムの場合、ビットかシンボル等の単位でインターリーピングにおける並び替えが行われる。

【0006】また、並び換えの方法には、バッファ等にデータを書き込み、それを読み取る方法と、インターリーピングによる順序の入替え情報をパターン（以下、「インターリーブ・パターン」と称する）として持ち、それを参照して並び替える方法がある。

【0007】次に、インターリーブ・パターンにより、ビット単位に並び替えを行った例を示す。

【0008】図 2 は、16 ビット系列のインターリーピングを行った例である。図 2 では、インターリーブ・パターン・テーブルを参照することによりビット単位のインターリーピングを行っている。図 2 では、インターリーピングが行われる入力 16 ビットの系列 67 は、インターリーブ・パターン・テーブル 68 に記憶されている順序にしたがって、入力系列内のビットの順序の入替えが行われる。また、そこに示されているインターリーブ・パターン・テーブルに示されている順序を、矢印のように縦方向の順に 0、8、4、12、2、・・・と読み出して、インターリーピング後のビット系列を出力する。

【0009】

【発明が解決しようとする課題】ところで、インターリーピングを行うインターリーブに対しては、

(1) 多様フレーム長（例えば、数千から 1 万種類）に対応すること。

(2) 少ないパラメータ数で生成できること。

(3) インターリーピングパターン生成の計算量が少ないことの 3 点の課題がある。

【0010】第 1 の課題、つまり、多様フレーム長に対応するために、単純に、すべてのフレームを用意すると、全てのフレーム長に対応するためのパラメータの数が膨大になってしまい、そのパラメータを記憶する所要メモリが膨大となるため非現実的である。さらに、フレーム長毎に個別に最適なパラメータを求めるための演算処理時間も膨大となるという問題がある。

【0011】また、この問題を解決するために、上記

(2) の課題に示すように、少ないパラメータ数で、インターリーブを生成できるようにする対策が考えられる。しかし、少ないパラメータ数で、インターリーブを生成できるようにするために、2 のべき乗のフレーム長についてインターリーブを作成し、そこからデータの間引きを行う従来の手法は、データを間引く分、それだけ

最適化のパラメータが増加し、全てのフレーム長で優れた特性が得られる保証が無くなってしまうという問題がある。例えば、あるフレーム長では特性がよくても、別のフレームでは特性が劣化するという問題がある。

【0012】それを改善するために、間引きデータの数を減らす方法が考えられる。

【0013】間引きデータの数を減らすことにより、第3の課題も解決される。この第3の課題に対する対策として、本出願人より、間引きを少なくしてかつ特性もよくなる方法（PCT出願/JP98/05027）が提案されている。しかし、この方法でも、インターリーブのパターン生成のための処理量（演算処理量）が多くなるという問題がある。

【0014】本発明は、上記問題に鑑みなされたものであり、インターリーブ方法、インターリーブ装置、ターボ符号化方法及びターボ符号化装置において、多様なフレーム長であっても、少ない演算量で効率的に系列のランダム化を実現することを目的とするものである。

【0015】

【課題を解決するための手段】請求項1に記載された発明は、素数Pをベースにした長さのブロックを複数個有するデータ系列を入力する第1の段階と、標数がPの有限体の元に所定の演算を行い、その順序を並べ替えて、順序入替えデータを生成する第2の段階と、該順序入替えデータを用いて、入力された前記データ系列のデータの順序を入替える第3の段階とを有することを特徴とするインターリーブ方法である。素数Pをベースにした長さのブロックを複数個有するデータ系列を用いることで、多様なフレーム長に細かく対応できるようになるとともに、少ない演算量で効率的に入力するデータ系列のランダム化が実現できる。

【0016】請求項2に記載の発明は、素数Pを生成又は記録する第1の段階と、入力系列を長さPのN個のブロック B_1, B_2, \dots, B_N に分割する第2の段階と、標数がPの有限体の元をその元のべき表現の指数部分の値の順に並べた系列を第1の順序入替えデータとして生成又は記録する第3の段階と、(P-1)とは互いに素な(N-1)個の整数 p_1, p_2, \dots, p_{N-1} を生成又は記録する第4の段階と、第1の順序入替えデータ系列を p_i 個飛びに巡回的に読み出して第iの順序入替えデータを得る処理を $i=1 \sim N-1$ だけ繰り返して第2～第Nの順序入替えデータを生成又は記録する第5の段階と、第1からNまでの順序入替えデータを用いてブロック B_1, B_2, \dots, B_N 中の順序を入替える第6の段階と、並び替えられたN個の各ブロックから予め決められた順序により各データを読み出す第7の段階とを有することを特徴とするインターリーブ装置である。入力系列をN個のブロック B_1, B_2, \dots, B_N に分割し、素数体を用いて、入力データの順序の入替

えを行うことで、多様なフレーム長に細かく対応できるようになるとともに、少ない演算量で効率的に入力するデータ系列のランダム化が実現できる。また、第5と第6の2段階でインターリーブを行うので、メモリ（バッファ）と演算量を削減することができる。

【0017】請求項3に記載の発明は、素数Pを生成又は記録する第1の段階と、入力系列を長さPのN個のブロック B_1, B_2, \dots, B_N に分割する第2の段階と、標数がPの有限体の元をその元のべき表現の指数部分の値の順に並べた系列を生成又は記録する第3の段階と、このべき表現に用いた原始元とは互いに素なN個の整数 q_1, q_2, \dots, q_N を生成又は記録する第4の段階と、第0の順序入替えデータ系列の各データに q_1 を法Pで加え得られた元のべき表現の指数部分の値の系列を第iの順序入替えデータとする処理を $i=1 \sim N$ まで繰り返して第1～第Nの順序入替えデータを生成する第5の段階と、第1からNまでの順序入替えデータを用いてブロック B_1, B_2, \dots, B_N 中のデータの順序を入替える第6の段階と、並び替えられたN個の各ブロックから予め決められた順序により各データを読み出す第7の段階とを有することを特徴とするインターリーブ方法である。入力系列をN個のブロック B_1, B_2, \dots, B_N に分割し、素数体を用いて、入力データの順序の入替えを行うことで、多様なフレーム長に細かく対応できるようになるとともに、少ない演算量で効率的に入力するデータ系列のランダム化が実現できる。また、第5と第6の2段階でインターリーブを行うので、メモリ（バッファ）と演算量を削減することができる。

【0018】請求項4に記載の発明は、素数Pを生成又は記録する第1の段階と、入力系列を長さ(P-1)のN個のブロック B_1, B_2, \dots, B_N に分割する第2の段階と、標数がPの有限体の元をその元のべき表現の指数部分の値の順に並べた系列の最後尾のデータを削除した系列を生成又は記録する第3の段階と、(P-1)とは互いに素な(N-1)個の整数 p_1, p_2, \dots, p_{N-1} を生成又は記録する第4の段階と、第1の順序入替えデータ系列を p_i 個飛びに巡回的に読み出して第iの順序入替えデータを得る処理を $i=1 \sim N-1$ だけ繰り返して第2～第Nの順序入替えデータを生成又は記録する第5の段階と、第1からNまでの順序入替えデータを用いてブロック B_1, B_2, \dots, B_N 中の順序を入替える第6の段階と、並び替えられたN個の各ブロックから予め決められた順序により各データを読み出す第7の段階とを有することを特徴とするインターリーブ方法である。これにより、間引き処理で処理すべきビット数を少なくすることができ、多様なフレーム長により柔軟に対応することができる。

【0019】請求項5に記載の発明は、素数Pを生成又は記録する第1の段階と、入力系列を長さ(P+1)の

N個のブロック B_1, B_2, \dots, B_N に分割する第2の段階と、標数がPの有限体の元をその元のべき表現の指数部分の値の順に並べた系列の最後尾のデータに前記素数を追加した系列を生成又は記録する第3の段階と、 $(P-1)$ とは互いに素な $(N-1)$ 個の整数 p_1, p_2, \dots, p_{N-1} を生成又は記録する第4の段階と、第1の順序入替えデータ系列を p_i 個飛びに巡回的に読み出して第 i の順序入替えデータを得る処理を $i = 1 \sim N-1$ だけ繰り返して第2～第Nの順序入替えデータを生成又は記録する第5の段階と、第1からNまでの順序入替えデータを用いてブロック B_1, B_2, \dots, B_N 中の順序を入替える第6の段階と、並び替えられたN個の各ブロックから予め決められた順序により各データを読み出す第7の段階とを有することを特徴とするインターリーピング方法である。これにより、間引き処理で処理すべきビット数を少なくすることができ、多様なフレーム長により柔軟に対応することができる。

【0020】請求項6に記載の発明は、前記第7の段階の予め決められた順序が、ターボ符号におけるエラーフロアの値を基準とすることを特徴とする請求項2ないし5のいずれか一項記載のインターリーピング方法である。エラーフロアの値を考慮して第7の段階で読み出す順序を決めるので、エラーフロアの発生を低く抑えることができる。

【0021】請求項7に記載の発明は、前記分割数Nを k (k は2以上の整数) 個予め決めておき、第6の段階で生成する第1からNまでの順序入替えデータを k 通り生成して、最も特性の良い分割数の順序入替えデータを用いることを特徴とする請求項2ないし5のいずれか一項記載のインターリーピング方法である。これにより、最も適した順序入替えデータを選択できるので、最適なインターリーピングを行うことができる。

【0022】請求項8に記載の発明は、素数Pをベースにした長さのブロックを複数個有するデータ系列を入力する第1の手段と、標数がPの有限体の元に所定の演算を行い、その順序を並べ替えて、順序入替えデータを生成する第2の手段と、該順序入替えデータを用いて、入力された前記データ系列のデータの順序を入替える第3の手段とを有することを特徴とするインターリーピング装置である。請求項1と同様の作用、効果が得られる。

【0023】請求項9に記載の発明は、素数Pを生成又は記録する第1の手段と、入力系列を長さPのN個のブロック B_1, B_2, \dots, B_N に分割する第2の手段と、標数がPの有限体の元をその元のべき表現の指数部分の値の順に並べた系列を第1の順序入替えデータとして生成又は記録する第3の手段と、 $(P-1)$ とは互いに素な $(N-1)$ 個の整数 p_1, p_2, \dots, p_{N-1} を生成又は記録する第4の手段と、第1の順序入替えデータ系列を p_i 個飛びに巡回的に読み出して第 i の順序入替えデータを得る処理を $i = 1 \sim N-1$ だけ繰り返して

て第2～第Nの順序入替えデータを生成又は記録する第5の手段と、第1からNまでの順序入替えデータを用いてブロック B_1, B_2, \dots, B_N 中の順序を入替える第6の手段と、並び替えられたN個の各ブロックから予め決められた順序により各データを読み出す第7の手段とを有することを特徴とするインターリーピング装置である。請求項2と同様の作用、効果が得られる。

【0024】請求項10に記載の発明は、素数Pを生成又は記録する第1の手段と、入力系列を長さPのN個のブロック B_1, B_2, \dots, B_N に分割する第2の手段と、標数がPの有限体の元をその元のべき表現の指数部分の値の順に並べた系列を生成又は記録する第3の手段と、このべき表現に用いた原始元とは互いに素なN個の整数 q_1, q_2, \dots, q_N を生成又は記録する第4の手段と、第0の順序入替えデータ系列の各データに q_i を法Pで加え得られた元のべき表現の指数部分の値の系列を第 i の順序入替えデータとする処理を $i = 1 \sim N$ まで繰り返して第1～第Nの順序入替えデータを生成する第5の手段と、第1からNまでの順序入替えデータを用いてブロック B_1, B_2, \dots, B_N 中のデータの順序を入替える第6の手段と、並び替えられたN個の各ブロックから予め決められた順序により各データを読み出す第7の手段とを有することを特徴とするインターリーピング装置である。請求項3と同様の作用、効果が得られる。

【0025】請求項11に記載の発明は、素数Pを生成又は記録する第1の手段と、入力系列を長さ $(P-1)$ のN個のブロック B_1, B_2, \dots, B_N に分割する第2の手段と、標数がPの有限体の元をその元のべき表現の指数部分の値の順に並べた系列の最後尾のデータを削除した系列を生成又は記録する第3の手段と、 $(P-1)$ とは互いに素な $(N-1)$ 個の整数 p_1, p_2, \dots, p_{N-1} を生成又は記録する第4の手段と、第1の順序入替えデータ系列を p_i 個飛びに巡回的に読み出して第 i の順序入替えデータを得る処理を $i = 1 \sim N-1$ だけ繰り返して第2～第Nの順序入替えデータを生成又は記録する第5の手段と、第1からNまでの順序入替えデータを用いてブロック B_1, B_2, \dots, B_N 中の順序を入替える第6の手段と、並び替えられたN個の各ブロックから予め決められた順序により各データを読み出す第7の手段とを有することを特徴とするインターリーピング装置である。請求項4と同様の作用、効果が得られる。

【0026】請求項12に記載の発明は、素数Pを生成又は記録する第1の手段と、入力系列を長さ $(P+1)$ のN個のブロック B_1, B_2, \dots, B_N に分割する第2の手段と、標数がPの有限体の元をその元のべき表現の指数部分の値の順に並べた系列の最後尾に前記素数を追加した系列を生成又は記録する第3の手段と、このべき表現に用いた原始元とは互いに素なN個の整数

q_1, q_2, \dots, q_N を生成又は記録する第 4 の手段と、第 0 の順序入替えデータ系列の各データに q_i を法 P で加え得られた元のべき表現の指数部分の値の系列を第 i の順序入替えデータとする処理を $i = 1 \sim N$ まで繰り返して第 1 ～第 N の順序入替えデータを生成する第 5 の手段と、第 1 から N までの順序入替えデータを用いてブロック B_1, B_2, \dots, B_N 中のデータの順序を入替える第 6 の手段と、並べ変えられた N 個の各ブロックから予め決められた順序により各データを読み出す第 7 の段階とを有することを特徴とするインターリーピング装置である。請求項 5 と同様の作用、効果が得られる。

【0027】請求項 13 に記載の発明は、前記第 7 の手段の予め決められた順序が、ターボ符号におけるエラーフロアーの値を基準とすることを特徴とする請求項 9 ないし 12 のいずれか一項記載のインターリーピング装置である。請求項 6 と同様の作用、効果が得られる。

【0028】請求項 14 に記載の発明は、前記分割数 N を k (k は 2 以上の整数) 個予め決めておき、第 5 の手段で生成する第 1 から N までの順序入替えデータを k 通り生成して、最も特性の良い分割数の順序入替えデータを用いることを特徴とする請求項 9 ないし 12 のいずれか一項記載のインターリーピング装置である。請求項 7 と同様の作用、効果が得られる。

【0029】請求項 15 に記載の発明は、請求項 1 ないし 5 の何れか一項記載のインターリーピング方法をターボ符号化装置の内部インターリーピング方法とするターボ符号化方法である。請求項 1 ないし 5 に記載の発明の作用、効果を有するターボ符号化方法を提供することができる。

【0030】請求項 16 に記載の発明は、入力ビット数が予め決められた数のビット数に足りない場合には、これに一致するようにビット数を増加させる段階と、符号化されたビット数を前記ビット数を増加させる前のビット数にまで削減する段階を含むことを特徴とする請求項 15 記載のターボ符号化方法である。これにより、間引き処理を行わずに演算量を低減することができる。

【0031】請求項 17 に記載の発明は、請求項 16 記載のターボ符号化方法において、ビットレピテーションを用いて前記ビット数を増加させることを特徴とするターボ符号化方法である。ビット数を増加させる一例を規定するものである。

【0032】請求項 18 に記載の発明は、複数の符号化器と、請求項 8 ないし 12 の何れか一項記載のインターリーピング装置とを具備するターボ符号化装置である。請求項 8 ないし 12 に記載の発明の作用、効果を有するターボ符号化方法を提供することができる。

【0033】請求項 19 に記載の発明は、入力ビット数が予め決められた数のビット数に足りない場合には、これに一致するようにビット数を増加させる手段と、符号化されたビット数を前記ビット数を増加させる前のビ

ット数にまで削減する手段を含むことを特徴とする請求項 18 記載のターボ符号化装置である。

【0034】請求項 20 に記載の発明は、請求項 19 記載のターボ符号化装置において、ビットレピテーションを用いて前記ビット数を増加させることを特徴とするターボ符号化装置である。

【0035】

【発明の実施の形態】次に、本発明の実施の形態について図面と共に説明する。

【0036】図 3 は、本発明の第 1 の実施例であるターボ符号器のブロック構成を示す。図 1 に示す従来のターボ符号器との差は、

- ①ビット追加処理部 21 の追加
- ②新規な構成のインターリーバ 22
- ③バンクチャ処理部 23 の追加の 3 点である。

【0037】以下、図 10 に示すフローチャートを参照して上記 3 つの各部を詳細に説明する。

(ビット追加処理) インターリーピングを行う前処理として、インターリーピングに適したビット数に調整する処理である (図 10 のステップ 101 ～ 103)。

【0038】ビット追加処理の具体例としては、一般の誤り訂正符号化を用いることができる。誤り訂正符号化の中でも、周期的にビットを繰り返すビットレピテーションは、その柔軟性と処理が平易であることから、好ましい例である。

【0039】ここで、符号器への入力ビット数を N_{IN} (図 3 の K に相当する) 仮定して、ビットレピテーションの処理方法について詳しく述べる。

(1) まず、 N_{IN} を 8 で割ってその値 n を求める。

(2) n 以上でかつ最も n に近い素数 P を求める。

(3) P の 8 倍と N_{IN} との差をとり、これを a とする。

(4) N_{IN} ビットを入力としてこれに a ビット (ダミービット) を追加する。

【0040】例えば、 N_{IN} が、650 の場合で説明する。

(1) $650 / 8 = 81.25$ であるから、 $n = 81.25$ が求まる。

(2) 81.25 以上で、最も 81.25 に近い素数は、図 4 に示すように、83 である。従って、 $P = 83$ が求まる。

(3) $83 * 8 = 664$ であるから、 $a = 14$ が求まる。なお、 $*$ は、乗算を表す (以下同じ)。

(4) 650 ビットの入力の場合は、14 ビットのダミービットを追加する処理を行う。

【0041】以上により得られた、 $(N_{IN} + a)$ ビット、つまり図 3 で言えば $(K + a)$ ビットは、上記の例の場合、必ず 8 で割れ、かつその商は素数となる。なお、8 を用いる理由は、後述するように、インターリーバ 22 におけるインターリーピングの第 1 ステージで扱

う 2 次元配列の行数が、本実施例の場合には 8 であるからである。従って、後置されるインターリーブ 22 の 2 次元配列の行数に応じて、8 以外にも 10 や 20 等の任意の値をとることができる。つまり、ターボ符号用のインターリーブの第 1 ステージの行数が 10 や 20 の場合には、上記 (1) ~ (3) での処理は 8 に代わって 10 や 20 の数値を用いる。

【0042】この点を考慮すれば、ビット追加処理部 21 の処理は、図 10 のステップ 101 で 2 次元配列の行数を決め、ステップ 102 で上述のようにして素数である列数を決め、行数と列数とをかけた値と入力データのビット数との差のビット数のダミービットをステップ 103 で入力データに付加するものである。

【0043】また、ここではビットレベションを用いる例について説明したが、ブロック符号化や畳込み符号化などもビット追加処理として適用可能である。さらに、ビット追加処理として、既知ビットを既知の場所に追加する方法も平易な方法として考えられる。

(インターリーブ) 本実施例で用いるインターリーブ 22 の 3 通りの構成例を説明する。

【0044】第 1 の構成例を図 5 に示す。このインターリーブは、第 1 ステージ 41 (図 10 のステップ 104 に相当する)、第 2 ステージ 42 (図 10 のステップ 105 ~ 108)、及び第 3 ステージ 43 (図 10 のステップ 109 に相当する) の 3 つのステージで構成される。

(1) 第 1 ステージ 41: 入力系列 40 (ビット追加処理部 21 の出力で、例えば 664 ビット) を、N 分割

(この例では、 $B_1 \sim B_8$ の 8 つのブロックに分割) して、2 次元配列 (バッファ) に書き込む。この例で行数は 8、列数は 83 である。

【0045】なお、前述したようにビット追加処理部 21 でのダミービットの付加により、2 次元バッファの行数は 8 で分割可能であり、且つ、列数は素数となる。

(2) 第 2 ステージ 42: 後述するようにして、各行のデータの順序を入替える (intra-permutation 処理)。

(3) 第 3 ステージ 43: 行を単位として行の順序を入替える (inter-permutation 処理)。例えば、予め学習 (学習の基準は自由距離を大きくすること) により決めた行間の交錯パターンを用いて、1 行

40

*

テーブル t_0 : 1, 2, 4, 8, 16, ... 42, 0 ... (1)

例えば、2 次元バッファの第 1 行に配列された入力デー

$A_0, A_1, A_2, A_3, \dots A_{82}$... (2)

とすれば、この第 1 行の並びは、この順序入替えテーブル t_0 を参照することで以下の通り入替えられる。例えば、順序入替えテーブル t_0 の 1 に対応する A_0 はそのままの位置に置かれ、2 に対応する A_1 もそのままの位置となり、4 に対応する A_2 は 4 番目に入替えられ、次の 8 に対応する A_3 は 8 番目に入替えられる。以下同様

50

* 【0046】このように第 1 から第 3 のステージの処理を行っていき、最後に縦方向 (列方向) に読み出して (図 10 のステップ 110)、インターリーブ処理された符号化系列 44 を得る。

【0047】以下、第 2 ステージでの処理について詳細に説明する。

【0048】第 2 ステージでデータの順序を入替える処理は、以下のステップを実行することで生成されるテーブルをアドレステーブルとして用いて、2 次元バッファに書き込まれた入力データを処理するものである。以下、ステップ順に説明する。

【0049】ステップ S1: 標数 P (図 5 の列数 83 に相当する) の有限体の原始元 g_0 を求め (図 10 のステップ 105)、その指数表現順のテーブル (有限体の元を真数で表現し、これを指数表現の順にならべたテーブル) t_0 を作成する。ただし、このテーブル t_0 はあらかじめ生成し記憶しておくことも可能である。

【0050】例えば、 $P=83$ の場合、図 6 に示すように、83 の原始元は 2 である。標数が 83 の有限体の元は、0, 1, 2, ... 82 である。有限体の元を真数で表現しこれを、法 2 の元で指数表現すると、

$2^0 \pmod{83}$ 、 $2^1 \pmod{83}$ 、 $2^2 \pmod{83}$ 、...、 $2^{82} \pmod{83} = 1, 2, 4, 8, 16, 32, 64, 45, 7, 14, \dots, 42, 0$

が得られる。

【0051】これをテーブルにすると、図 7 (B) のテーブル t_0 が得られる。図 7 (B) において、縦軸方向と横軸方向の数値で指数を表す。例えば、縦軸の 1 と横軸の 6 で指数 16 を示す。 2^2 の $\pmod{83}$ 演算結果は 4、 2^{16} の $\pmod{83}$ 演算結果は 49 である。なお、 2^{82} の場合は 0 とする。

ステップ S2: テーブル t_0 を、2 次元バッファの第 1 行 (図 10 のステップ 106 で行番号を示すパラメータ 1 を 1 に設定した場合) のデータの順序を入替えるために参照する順序入替えテーブルとする。つまり、順序入替えテーブル t_0 に規定された数値は、入替え後の入力データの位置を示している。図 7 (B) に示すように、順序入替えテーブル t_0 は左上から順序に以下の並び (パターン) を有する。

【0052】

タが ... (1)

タが ... (2)

であり、最後の 0 に対応する A_{82} は 0 なのでそのままの位置となる。この処理が図 10 のステップ 107 である。

【0053】従って、(2) の列のデータは入替えられて次のようになる。

【0054】

$A_0, A_1, A_{72}, A_2, A_{27}, A_{76}, A_8, \dots, A_{82}$

(3)

ステップS3: 有限体の標数から1を引いた数と互いに素な数を(行数-1)求める。上述の例では $P=83$ 、行数は8なので、 $82 (=P-1=83-1)$ と互いに素な数を7($=8-1$)個求め、 $p_1, p_2, p_3, p_4, p_5, p_6, p_7$ とする。例えば、 $P-1 (82=41 \times 2)$ とは、互いに素な $(N-1)$ 個の整数($N-1=7$) $p_1, p_2, p_3, p_4, p_5, p_6, p_7$ は、例えば、3、5、7、11、13、17、19 (1、2を除く)である。

テーブルt0: 1、2、4、8、16、 \dots 、42、0 \dots (1)

の値を $p_1 (=3)$ 個飛びに読みだして、次のテーブルt1を得る。

テーブルt1: 1、16、7、 \dots

なお、ここでの処理は、 g_0 と異なる原始元 g_1 を求める、その原始元を用いて指数表現のテーブルを生成することによっても実現可能である(数学的に等価である)。但し、 $g_1 = (g_0)^{p_1} \pmod{83}$ 。
ステップS5: そして、テーブルt1を第2行のデータの順序を入替えるために参照する順序入替えテーブルとする。

ステップS6: 同様に、 $p_2, p_3, p_4, p_5, p_6, p_7$ を用いて、ステップS4とステップS5の処理を繰り返すことで、系列t2からt7を生成し、それぞれ2次元パツファの第3行から第8行のデータの順序を入替えるために参照する順序入替えテーブルとする。つまり、ステップ106~108は以下の通り記述できる。

【0056】まず、次の条件を満たす素数 l_i ($i=2 \sim r$, r は行数)を求める。

【0057】(i) $(83-1, l_i) = 1$ (82と l_i は互いに素)

(ii) $l_i > 6$

例えば、 $r=8$ の場合、求める素数は $l_2 \sim l_8$ で、図6のテーブルから、7、11、13、17、19、23、19となる。そして、テーブルt0の値を l_i 個飛びに巡回的に読み出す(最後の0は除く)ことで、順序入替えテーブルt2~t7を作成する。

ステップS7: 第1行から第8行のデータの順序を入替えるための順序入替えテーブル(t0~t7)により、ブロックB1、B2、 \dots B8のデータの順序を入替える。つまり、ブロックB1のデータの順序を順序入替えテーブルt0で入替える。ブロックB2のデータの順序を順序入替えテーブルt1で入替える。以下、同様にして、ブロックB8のデータの順序を順序入替えテーブルt7で入替える。なお、図10では1行毎に順序入替えテーブルを作成して入替え処理を行う手順であるが、上記の通り、8つの順序入替えテーブルを作成した後

T0: 1、2、4、8、16、 \dots 42、0 \dots (5)

*ステップS4: ここで、図10のステップ108で、 l_i の値が2次元配列の行数より小さいかどうか判断され、YESの場合はステップ107に戻る。ここでは、 $l_i=2$ となる。そして、第2行のデータの順序を入替えるための順序入替えテーブルを以下のようにして作成する。順序入替えテーブルt0の値を p_1 個飛びに巡回的に読み出し、これをならべた系列をt1とする。例えば、 p_1 が3の場合は、前述した順序入替えテーブルt

※ 0
※【0055】

\dots (4)

に、各行の入替え処理を行うことでもよい。

【0058】なお、第2ステージの処理については、上記順序入替えテーブルをあらかじめ作成しておき、このテーブルを参照する方法を用いることでも実現可能である。

【0059】本発明に用いるインターリーバ22の第2の構成例を、図8を参照して説明する。第2の構成例は、第2のステージを除き前述した第1の構成例と同じである。

【0060】本構成例の第2ステージでのデータ順序の入替え処理は、以下の処理で生成されるテーブルをアドレステーブルとして用いて実現する。

ステップS11: 標数83の有限体の原始元 g_0 を求める、その指数表現順のテーブル(有限体の元を真数で表現しこれを指数表現に順にならべたテーブル)T0を作成する。ただし、このテーブルはあらかじめ生成し記憶しておくことも可能である。ステップS11は、前述した第1の構成例のステップS1と同じである。従って、テーブルT0は、図7(B)に示すテーブルt0と同じテーブルとなる。

ステップS12: 原始元 g_0 と互いに素な数を8(つまり、2次元パツファの行数に等しい数)個求め、 $q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8$ とする。例えば、素数が83の場合、 $P=83$ 、原始元=2であるので、この原始元とは互いに素な8個の整数は、例えば、

3、5、7、11、13、17、19、21 (1、2を除く)

となる。

ステップS13: ステップS12で得られた、テーブルT0の各データに q_i を加算($\pmod{83}$)し、得られた値(真数)を指数表現に変換してT1テーブルを作成し、第1行の順序入替えテーブルとする。

【0061】つまり、

であるから、 $q_1 = 3$ の場合は、法 83 のもとで、それ
を得る。

【0062】さらに、これを、指数表現に変換する。図 *
2、27、8、24、...7、72

が得られる。これが順序入替えテーブル T1 となる。

ステップ S14：同様に、 q_2 、 q_3 、 q_4 、
 q_5 、 q_6 、 q_7 、 q_8 を用いて、ステップ S13 の処
理を繰り返すことで、テーブル T2 からテーブル T8 を
作成し、第 2 行から第 8 行までのデータの順序を入替える
ための順序入替えテーブルとする。

ステップ S15：第 1 から第 8 の順序入替えテーブル
(T1~T8) により、 B_1 、 B_2 、... B_8 のブ
ロックのデータの順序をそれぞれ入替える。

【0063】なお、第 2 ステージの処理については、以
上のテーブルをあらかじめ作成しておき、このテーブル
を参照する方法を用いることでも実現可能である。

【0064】次に、本発明に用いるインターリーブ 22
の第 3 の構成例を、図 9 を参照して説明する。

【0065】図 9 において、例えば、1140 ビットの
入力系列 80 を 72×16 の 2 次元配列でインターリーブ
600 に書き込んだ後、 72×16 インターリーブ 600
の行毎に 16 ビットずつ読み出す。そして、第 1 行目は
 4×4 インターリーブ 610 で、第 2 行目は 6×3 イン
ターリーブ 620 で、第 3 行目は 8×2 インターリーブ 630
等のように、行毎にインターリーブの形を変えて、イン
ターリーピングを行う。しかし、行毎に、全て同じ形の
インターリーブを用いてもよい。また、インターリー
ピングの一部を同じ形のインターリーブを用いてもよい。

【0066】このようにインターリーピングしたデータ
を、縦方向に読みだし (0、16、32、48、...
・)、出力のデータ系列 90 を得ることができる。

【0067】なお、最終行が 4 ビットしかないため、図
9 では、最終行に、 4×1 のインターリーブを用いた。
但し、 4×4 、 2×2 等のインターリーブでもよい。読
出し時に、通常通り、1136、1137、1138、
1139 と読みだすことも可能であるが、図 9 では、逆
順、つまり、1139、1138、1137、1136
と読みだした。

【0068】また、最終行が 4 ビットしかないため、最
終行を除いて (つまり、71 行だけ) 読みだし、最終行
のデータは、その後、所定の間隔を置いて、入れ込んで
もよい。

【0069】以上説明したインターリーブの第 1~第 3
の構成例のいずれかを用いて符号化データを生成する。
そして、図 10 のステップ 110、つまり図 5 や図 8 の
第 3 ステージ 43 を行う。

【0070】ここで、前述したインターリーブ 22 の第
1 及び第 2 の構成例において、図 10 のステップ 110
での処理を工夫することで、ターボ符号のエラーフロア

それに 3 を足して、

... (6)

*7 (A) は、図 7 (B) の逆演算であるので、それを用
いると、

... (7)

一発生の原因となるパターンが生じないようにすることが
できる。

【0071】図 11 は、エラーフロアを説明するため
のグラフである。エラーフロアとは、 S/N 比が向上
してもビットエラー率 (BER: Bit Error
Rate) の改善があまり得られない現象を示す。図 1
1 では、BER が 10^{-7} から 10^{-8} でエラーフロア
が発生し始め、それ以下では改善があまり見られな
い。

【0072】この現象を考慮して、2 次元配列 (バッ
フア) からのデータの読出し順序は固定ではなく、複数の
順序の読出しが可能である。つまり、並べ変えられた N
個の各ブロックから、各データを読み出す予め決められ
た順序は、ターボ符号におけるエラーフロアの値を基
準に決められることにより、ターボ符号におけるエラー
フロアの発生を低く抑えることができる。例えば、1
0 個のブロック (第 1 から第 10 ブロック) に分けた場
合には、それを読みだす順序を、10、9、8、7、
6、5、4、3、2、1 とし、20 個のブロック (第 1
から第 20 ブロック) に分けた場合には、それを読みだ
す順序を、19、9、14、4、0、2、5、7、1
2、18、16、13、17、15、3、1、6、1
1、8、10 とする。また、20 個の場合には別の順
序、19、9、14、4、0、2、5、7、12、1
8、10、8、13、17、3、1、16、6、15、
11 も適用できる。

【0073】このように、幾つかの順序での読み出し順
序の内、ターボ符号におけるエラーフロアの発生を低
く抑えるものを選択する。10 個のブロックに分けた場
合の例のように、読出し順序を単に逆順にする方法は平
易で効果も高い。

(バンクチャ処理) 従来法のターボ符号器では、図 1 に
示すように、入力 K ビットに対して ($3 \times K + T1 + T2$)
ビットの出力 (符号化) ビットが得られるので、本
発明でも同じビット数の符号化ビットを出力する (こ
こで、T1 は RSC1 のテールビット数、T2 は RSC2 の
テールビット数とする)。

【0074】図 3 に示すビット追加処理部 21 でのダミ
ービットの追加処理により、入力ビット数が N (図 3 の
K に相当) から ($N + a$) にビット数が増加しているの
で、ビット追加処理しない場合と比較して、全体で ($3 \times a$)
ビットだけ余分となる。そこで、($3 \times a$) ビッ
トを削減するためにバンクチャリング処理部 23 でバン
クチャリングを行う。ターボ符号用のバンクチャリン
グとしては、冗長ビットのみを周期的に削除する方法が

一般的であり、本発明においてもこれが適用可能である。この結果、バンクチャリング処理部 23 の出力は、入力 k ビットに対して $(3 * K + T1 + T2)$ ビットの符号化出力となる。

【0075】次に、本発明の第 2 の実施例を説明する。

【0076】図 12 は、本発明の第 2 の実施例のターボ符号器の構成を示すブロック図である。図 12 において、図 3 に示す構成要素と同一のものには同一の参照番号を付してある。図 12 の構成は、ビット追加処理部 21 をインターリーブ 22 の前段のみに置いたものである。つまり、符号化系列 $X1$ は情報源からの入力データ系列そのものであり、 $RSC12$ は情報源からの入力データ系列をそのまま処理する。また、ビット追加処理部 21 で追加したダミービットを削除するために、ブルーニング（間引き処理）部 123 をインターリーブ 22 と $RSC13$ との間に設けてある。インターリーブ 22 は、前述した第 1 ～ 第 3 の構成例のいずれかで構成できる。しかしながら、ここでは、新たに第 4 の構成例を用いた場合を説明する。第 4 の構成例は、第 1 及び第 2 の構成例をベースに若干の変更を加えたものである。この変更とは、図 4 のテーブルを参照して素数を求める場合の工夫、換言すれば 2 次元配列の列数を決める場合の工夫である。この点について、以下に説明する。

【0077】まず、ここでのビットレピティション処理は次の通りである。ただし、2 次元配列の行数を 8 とする。

(1) まず、 N_{IN} を 8 で割ってその値 n を求める。

(2) n 以上でかつ最も n に近い素数 P と (素数 - 1) 及び (素数 + 1) の中から、 n 以上でかつ最も n に近い数を求める。

(3) P の 8 倍と N_{IN} との差をとり、これを a とする。

(4) N_{IN} ビットを入力としてこれに a ビット（ダミービット）を追加する。

【0078】例えば、 N_{IN} が、660 の場合で説明する。

(1) $660 / 8 = 82.5$ （商は 82、余り 4）であるから、 $n = 82.5$ が求まる。

(2) 82.5 以上でかつ 82.5 に最も近い素数は 83 であり、また (素数 - 1) = 82、(素数 + 1) = 84 なので、82.5 以上でかつ 82.5 に最も近い数は 83 である。従って、 $P = 83$ が求まる。

(3) $83 * 8 = 664$ であるから、 $a = 664 - 660 = 4$ が求まる。

(4) 660 ビットの入力の場合は、4 ビットのダミービットを追加する処理を行う。

【0079】以上により得られた $(N_{IN} + a)$ ビット、つまり図 3 で言えば $(K + a)$ ビットは、上記の例の場合、必ず 8 で割れ、かつその商は素数、(素数 - 1) 又は (素数 + 1) のいずれかとなる。

【0080】上記計算の商が素数に一致した場合、順序入替えテーブルの生成方法については、例えば図 5 を参照して説明した通りである。商が (素数 - 1) 又は (素数 + 1) に一致した場合、つまり図 13 に示すように 2 次元バッファの列の数が例えば 82 や 84 になった場合、これらの順序入替えテーブルは、列数 83 の順序入替えるための順序入替えテーブル $t0$ を用いることはできない。列数 82 や 84 の順序入替えテーブルは、列数 83 の順序入替えテーブル $t0$ を次の通り処理して作成する。

【0081】図 14 (A) は、列数 83 の順序入替えテーブル $t0$ を示す。これは、図 7 (B) の順序入替えテーブル $t0$ と同一である。列数 82 の順序入替えテーブル ($t0-1$ とする) は、図 14 (B) に示すように、列数 83 の順序入替えテーブル $t0$ の最後の 0 を削除することで得られる。これを一行に展開すると、以下のパターンとなる。

【0082】テーブル $t0-1$: 1、2、4、8、16、...、42

ただし、テーブル $t0-1$ の要素の範囲は 1 から 82 となっているため、全ての要素から 1 を引く（従って、要素の範囲を 0 から 81 として）処理を適用し、順序入替えテーブルとして用いる。また、列数 84 の順序入替えテーブル $t0+1$ は、図 14 (C) に示すように、列数 83 の順序入替えテーブル $t0$ の最後の 0 に素数 P 、つまり 83 を付加することで得られる。これを一行に展開すると、以下のパターンとなる。

【0083】テーブル $t0+1$: 1、2、4、8、16、...、42、0、83

そして、前述したステップ 106 ～ 108 の処理を実行することで、順序入替えテーブル $t0$ 、 $t0-1$ 、 $t0+1$ のそれぞれに対し第 2 行から第 8 行までの順序入替えテーブル $t1 \sim t7$ 、 $t1-1 \sim t7-1$ 、 $t1+1 \sim t7+1$ を作成する。なお、前述したように、これらのテーブルを予め作成し登録することとしても良い。

【0084】以上のようにして、入力データ系列のビット数とインターリーブ 22 での処理するビット数との差を小さくすることができ、その後のブルーニング部 123（図 12）での間引き処理を軽減することができ、多様なフレーム長に容易に対応することができ、優れたインターリーブ用のパターンを得ることができる。

【0085】なお上記の処理を図 3 の構成におけるインターリーブ 22 に適用しても良い。

【0086】以上説明した第 1、第 2 の実施例の構成では、入力データ系列を予め決めておいた単一の分割数で複数のブロックに分割するものであった。しかしながら、入力データ系列の分割数 N を k 個 (k は 2 以上の整数) とし、 k 個のインターリーブを作成しておき、最も特性の良い分割数のインターリーブを選択することとしても良い。

【0087】 $k=2$ とし、10と20の場合を考える。インターリーブ22への入力ビットが640ビットとする。ブロック数が10の場合には64ビット長のブロックが10個となり、これをもとに作成したインターリーブの順序入れ替えテーブル（パターン）を#1とする。一方、ブロック数が20の場合には、32ビット長のブロックが20個となり、これをもとに作成したインターリーブの順序入れ替えテーブル（パターン）を#2とする。インターリーブのパターン#1と#2とは異なり、ビット誤り率やフレーム誤り率などの特性の良いものを選択する。入力ビットが異なると、それに適したブロック数が異なる。すなわち、入力ビット数に応じてブロック数を選択的に変化させることにより、特性を向上させることができる。

【0088】

【発明の効果】以上説明したように、本発明によれば、素数体を用いることで多様なフレーム長にこまめに対応することができるとともに、少ない演算量で効率的に系列のランダム化を実現することができる。

【図面の簡単な説明】

【図1】従来のターボ符号及び復号の例を説明するための図である。

【図2】従来の16ビット系列のインターリーブを行った例を説明するための図である。

【図3】本発明の第1の実施例によるターボ符号器を説明するための図である。

【図4】200までの素数表である。

【図5】本発明のインターリーブの第1の構成例を説明

するための図である。

【図6】150以下の素数とその最小の原始根の表である。

【図7】データの順序を入替える順序入れ替えテーブルの一例を示す図である。

【図8】本発明のインターリーブの第2の構成例を説明するための図である。

【図9】本発明のインターリーブの第3の構成例を説明するための図である。

10 【図10】本発明のターボ符号器の動作を示すフローチャートである。

【図11】エラーフロアーを説明するための図である。

【図12】本発明の第2の実施例によるターボ符号器を示す図である。

【図13】本発明のインターリーブの第4の構成例を説明するための図である。

【図14】図13に示す第4の構成例で用いられる順序入れ替えテーブルを示す図である。

【符号の説明】

- 20 11、22 インターリーブ
12、13 RSC
21 ビット追加処理
23 バンクチャリング
40、80 入力ビット系列
41 第1のステージ
42 第2のステージ
43 第3のステージ
44、90 出力データ系列

【図4】

200までの素数表

(1)	101
2	103
3	107
5	109
7	
11	113
13	127
17	131
19	137
23	139
29	149
31	151
37	157
41	163
43	167
47	173
53	179
59	181
61	191
67	193
71	197
73	199
79	
83	
89	
97	

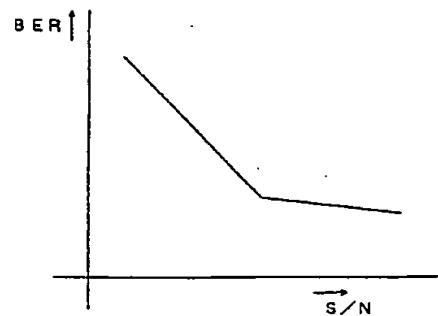
【図6】

150以下の素数とその最小の原始根の表

p	q	p	q	p	q
2	1	53	2	127	3
3	2	59	2	131	3
5	3	61	2	137	3
7	3	67	2	139	2
11	2	71	7	149	2
13	2	73	5		
17	3	79	3		
19	2	83	2		
23	5	89	3		
29	2	97	5		
31	3	101	2		
37	2	103	5		
41	6	107	2		
43	3	109	6		
47	5	113	3		

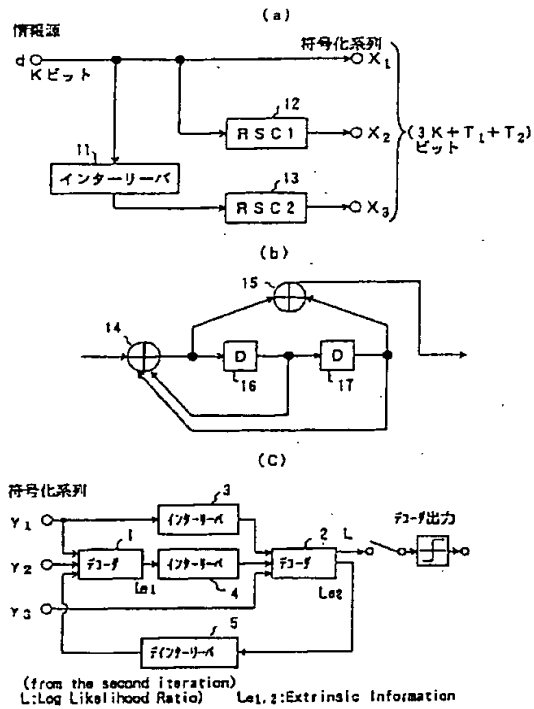
【図11】

エラーフロアーを説明するための図



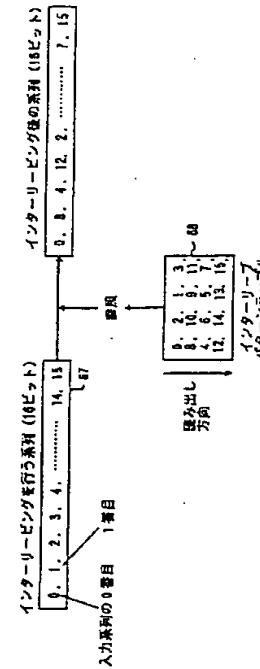
【图 1】

従来のターボ符号及び復号の例を説明するための図



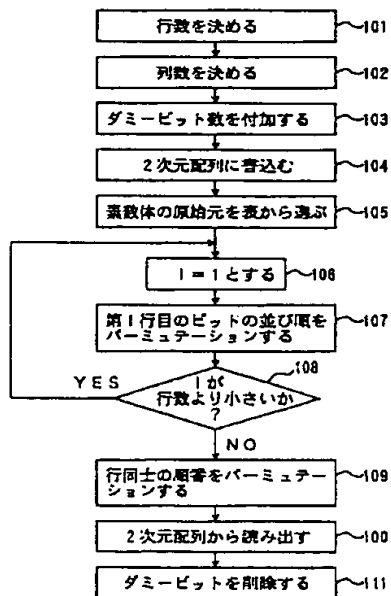
【図 2】

従来の16ビット系列のインタリービングを行った例を説明するための図



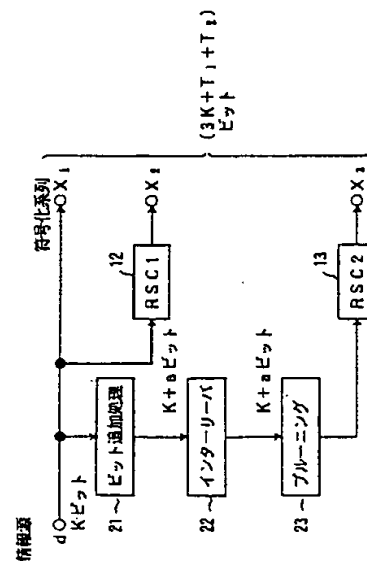
【圖 10】

本発明のターボ符号器の動作を示すフローチャート



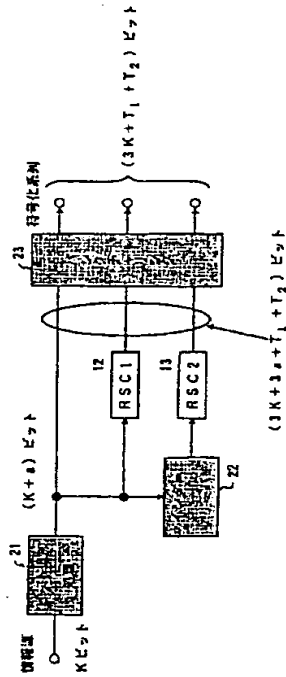
【图 12】

本発明の第 2 の実施例によるターボ符号器を示す図



【図 3】

本発明の第 1 の実施例によるターボ符号器を説明するための図



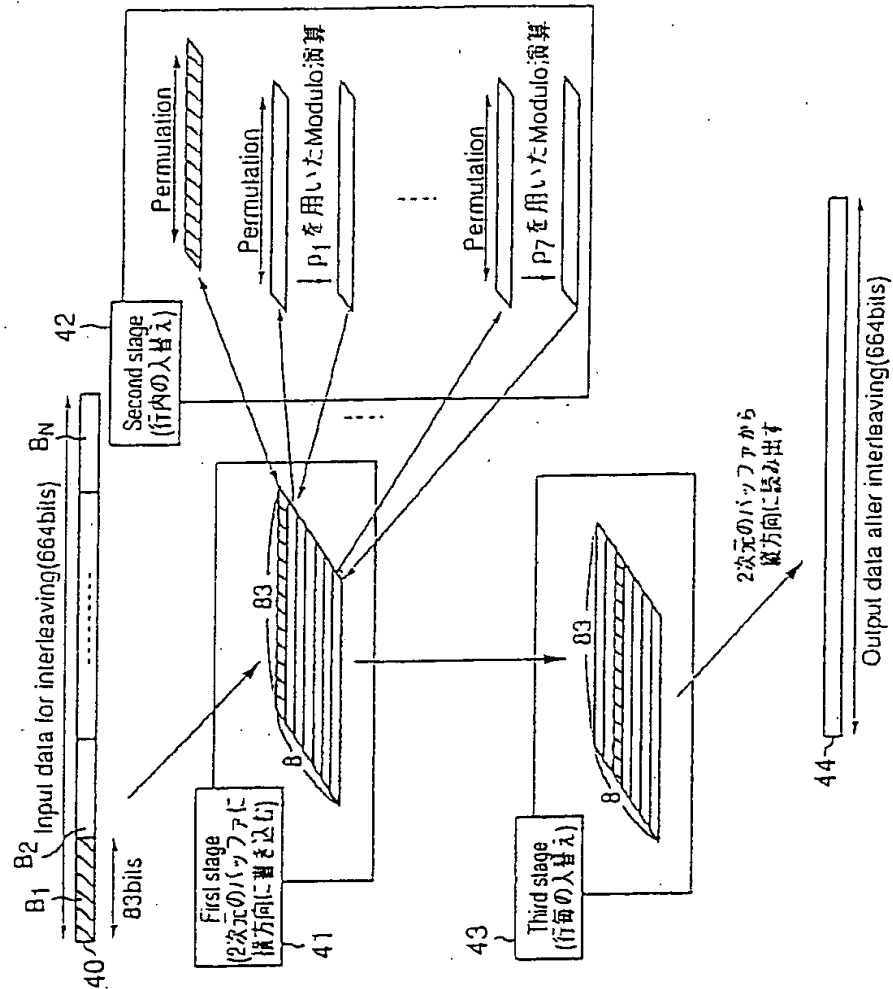
【図 7】

データの順序を入替える順序入替えテーブルの一例示図

(A)									
N	0	1	2	3	4	5	6	7	8
	0	82	24	74	25	60	14	77	12
	1	28	29	16	30	40	61	78	47
	2	35	33	49	34	59	79	54	10
	3	19	65	18	75	57	35	64	6
	4	38	53	41	26	53	7	51	3
	5	10	68	17	76	58	6	61	11
	6	30	32	48	37	50	7	52	4
	7	13	63	15	73	56	5	62	9
	8	44	55	43	27	54	8	63	1
	9	21	42	31	21	44	9	64	5
(B)									
b	1	0	1	2	3	4	5	6	7
	0	1	28	56	4	8	16	32	64
	1	25	74	61	29	11	22	44	88
	2	37	74	61	29	11	22	44	88
	3	40	80	67	32	14	28	56	112
	4	41	82	69	33	15	30	60	120
	5	49	89	77	41	21	42	84	168
	6	50	90	78	42	22	44	88	176
	7	51	91	79	43	23	46	92	184
	8	52	92	80	44	24	48	96	192
	9	53	93	81	45	25	50	100	200

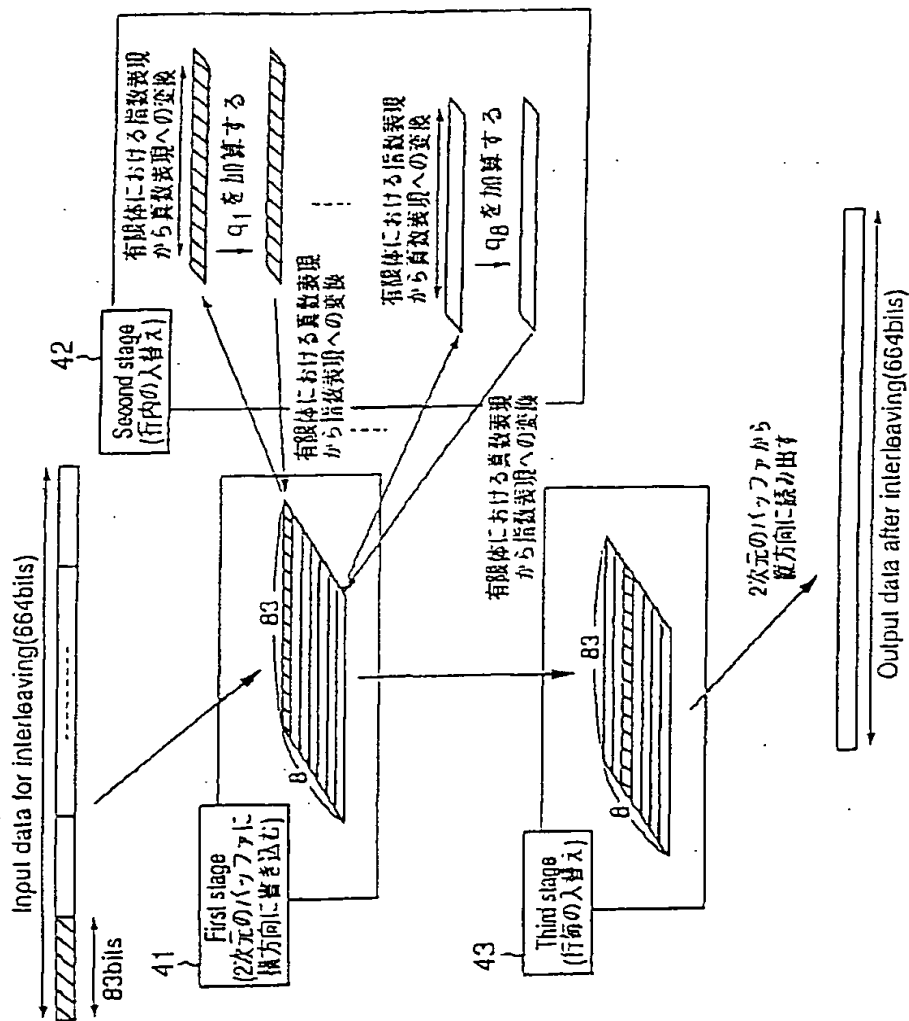
【図5】

本発明のインターリーバの第1の構成例を説明するための図



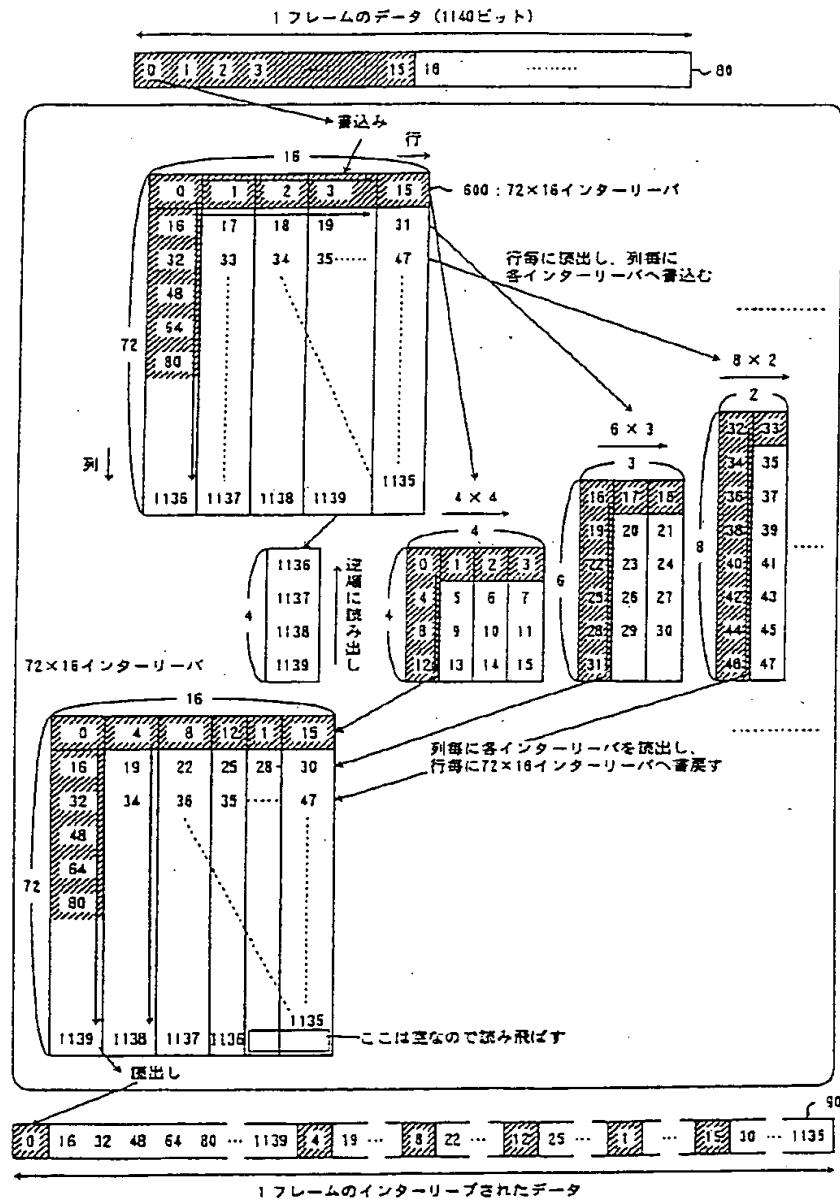
【図8】

本発明のインターリーブの第2の構成例を説明するための図



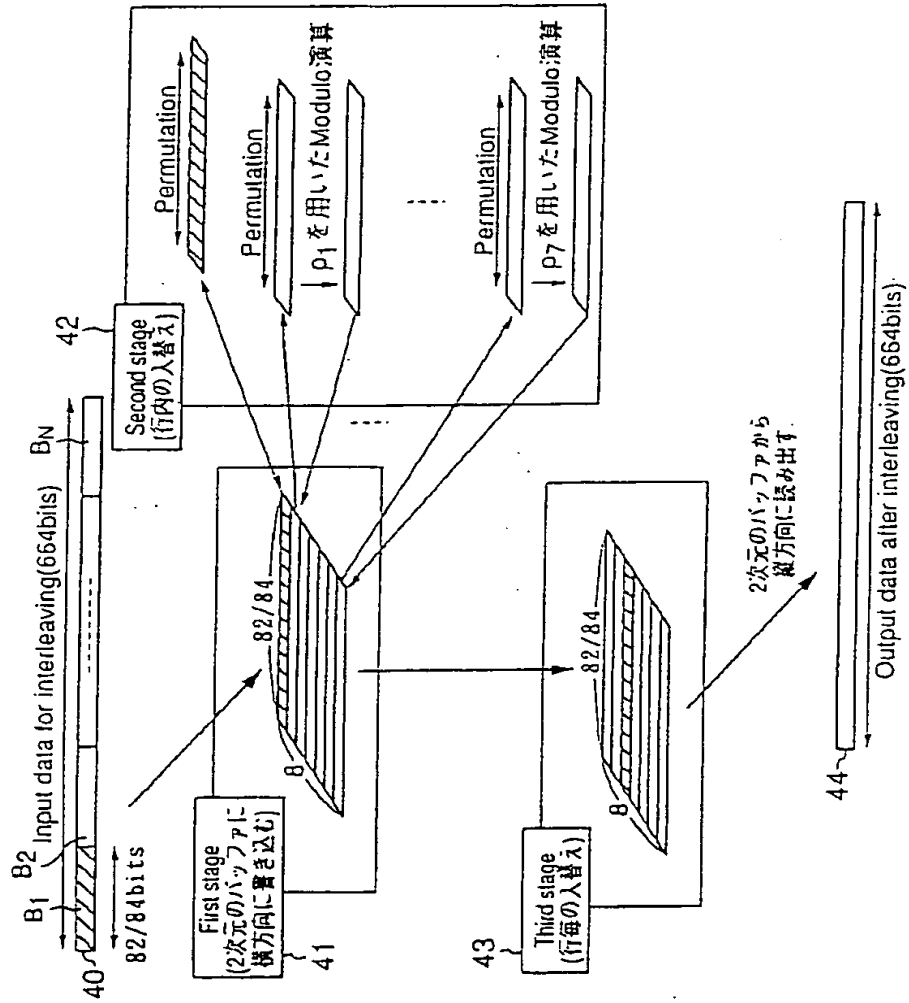
【図9】

本発明のインターリーブの第3の構成例を説明するための図



【図13】

本発明のインターリーブの第4の構成例を説明するための図



【図 14】

図 13 に示す第 4 の構成列で用いられる順序入替えテーブルを示す図

(A)									
1	0	1	2	3	4	5	6	7	8
0	1	2	3	4	5	6	7	8	9
1	2	3	4	5	6	7	8	9	0
2	3	4	5	6	7	8	9	0	1
3	4	5	6	7	8	9	0	1	2
4	5	6	7	8	9	0	1	2	3
5	6	7	8	9	0	1	2	3	4
6	7	8	9	0	1	2	3	4	5
7	8	9	0	1	2	3	4	5	6
8	9	0	1	2	3	4	5	6	7

(B)									
1	0	1	2	3	4	5	6	7	8
0	1	2	3	4	5	6	7	8	9
1	2	3	4	5	6	7	8	9	0
2	3	4	5	6	7	8	9	0	1
3	4	5	6	7	8	9	0	1	2
4	5	6	7	8	9	0	1	2	3
5	6	7	8	9	0	1	2	3	4
6	7	8	9	0	1	2	3	4	5
7	8	9	0	1	2	3	4	5	6
8	9	0	1	2	3	4	5	6	7

(C)									
1	0	1	2	3	4	5	6	7	8
0	1	2	3	4	5	6	7	8	9
1	2	3	4	5	6	7	8	9	0
2	3	4	5	6	7	8	9	0	1
3	4	5	6	7	8	9	0	1	2
4	5	6	7	8	9	0	1	2	3
5	6	7	8	9	0	1	2	3	4
6	7	8	9	0	1	2	3	4	5
7	8	9	0	1	2	3	4	5	6
8	9	0	1	2	3	4	5	6	7

THIS PAGE BLANK (USPTO)